安田女子大学紀要　42，231–247　2014.

完全なビジュアルプログラミング言語 BBB の設計と開発

山　下　明　博

Design and Development of Perfect Visual Programming Language BBB

Akihiro YAMASHITA

は　じ　め　に

コンピュータを動作させるためには，ソフトウェアを開発する必要がある。この，ソフトウェア開発のために使用されるプログラミング言語は，コンピュータが出現した当初は機械語，アセンブラ言語といった低水準言語が使われていたが，生産性，保守性を向上させるために，次第に高水準言語が使われるようになった。また，20世紀末までは，プログラミング言語は，大半が，テキストエディタを使うテキストプログラミング言語であったが，コンピュータにおいて GUI（Graphical User Interface）が多用されるようになると，マウスを使ってビジュアルに画面上の部品を設計できるビジュアルプログラミング言語が登場するようになった。特に，ビジュアルプログラミング言語の代表格である Visual BASIC は，マウスを使って Windows アプリケーションソフトを極めて容易に開発できたため，多くのユーザの支持を集めることとなった。ただし，Visual BASIC は，イベント駆動の部分の開発をビジュアルに行うことはできないため，完全なビジュアル言語とは言い難く，半ビジュアル言語の地位に留まっている。

このような現状を踏まえ，筆者は，プログラミング言語における完全なビジュアル化を実現することを目標に，BBB（Box Based BASIC）というプログラミング言語を設計・開発した。BBB は，画面上の部品を設計する部分をビジュアル化するだけでなく，これまではテキストエディタで記述していたロジックの部分のプログラミングも含めた全ての操作を，「ボックス」と呼ぶ長方形に基づいてビジュアル化することを可能にしたプログラミング言語である。

また，BBB は，複数のプログラミング言語のソースを出力することにより，プログラミングをすべてビジュアル化し，さらに多くの環境でソフトウェアを動作させることも可能としている。

本論文は，筆者が設計し，開発した BBB について，その技術的特徴と，具体的なプログラミング手順を示すことにより，完全なビジュアルプログラミング言語の有用性を明らかにするものである。

Ⅰ．　プログラミング言語の変遷

1.　低水準言語から高水準言語へ

コンピュータを動作させるためには，ハードウェアを制御するソフトウェアをプログラミングする必要がある。コンピュータが登場したばかりのころは，機械語を用いてプログラミングを行

い，ソフトウェアが開発されていた。当初，コンピュータの速度は遅く，記憶容量も小さく，プログラミングを行う手段は機械語しか存在しなかった。しかし，16進数で表記された機械語によるプログラミングは，人間にとってわかりにくいため，極めて生産性が低く，しかも可読性も低いため，デバッグ作業が大変困難であるという欠点を有していた。

このような機械語の問題点を少しでも解決する目的で，アセンブラ言語が開発された。アセンブラ言語は，機械語とほぼ1対1に対応する低水準言語ではあるものの，16進数ではなく，ニーモニックと呼ばれる命令語でプログラムを記述するようにしたことにより，可読性が向上し，機械語によるプログラミングに比べ，大きく生産性・保守性が向上した。ただし，機械語と密接に結びついているため，少し規模が大きくなると，プログラミングが困難になるという問題を抱えていた。

その後，プログラミングの生産性・保守性を高めるためには，低水準言語とは異なり，一層人間が理解しやすい言語を設計・開発すべきであるというアイデアが提案され，それを実現したのが，高水準言語と呼ばれるプログラミング言語である。コンピュータの歴史の中でも，比較的早い時期から登場した高水準言語には，科学技術計算に強く大型汎用コンピュータで利用されたFORTRAN，事務計算に強いとされたCOBOL，初心者でもすぐにプログラミング可能でありパソコンでしばしば利用されたBASICなどがある。

他方，ソフトウェア工学の発展に伴い，構造化プログラミング，オブジェクト指向プログラミングといった，生産性，可読性を高める手法が提唱され，それらの手法を実装したPASCAL，C，C++，Javaなど，多種多様な高水準言語が登場することとなった。これらの言語は，大規模なソフトウェアの開発にも使用され，ソフトウェアの生産性・保守性向上に大きな役割を果たしてきた。

2. テキストプログラミング言語からビジュアル言語へ

機械語，アセンブラ言語といった低水準言語や，FORTRAN，COBOL，BASIC，PASCAL，C，C++，Javaといった高水準言語に共通するのは，これらの言語を用いてプログラミングに，テキストエディタを使用するという点である。これを，テキストプログラミング言語と呼ぶことにする。

それに対し，視覚的な操作により，プログラミングを行うという言語が登場するようになった。これを，ビジュアル言語と呼ぶことにする。

最も成功したビジュアルプログラミング言語が，Visual BASIC言語[1)]である。20世紀末に，コンピュータにおいて，Windowsに代表されるGUI（Graphical User Interface）が多用されるようになり，Windows上で動作するプログラミングを開発しようとするプログラマが急増した。しかし，C++というテキストプログラミング言語によるWindowsプログラミングを行うためには，MFC（Microsoft Foundation Class）というWindowsで動作するプログラムを作成するためのクラスライブラリを理解する[2)]ことが必要であった。MFCは，従来のテキストベースのMS-DOSの世界には存在しない考え方であり，多くのプログラマにとって必要不可欠な概念であったにもかかわらず，MFCを理解することは極めて困難であった。

そのような状況下において，次第に注目を浴び，急速にユーザを増やしていったのが，Visual BASICというプログラミング言語である。Visual BASICは，古くからパソコンで利用されてきたBASIC言語との親和性を有すると共に，マウスを使ってビジュアルに，ボタンやチェックボッ

クスといった Windows 部品を容易に作成できる手軽さも有しており，初心者でもすぐにプログラミングすることが可能であった。それ故に，Visual BASIC は多くのユーザに受け入れられることとなった。図 1 に，Visual BASIC 6.0 での Windows 部品作成例を示す。左側に表示されたツールボックスから，作成したい部品の種類を選び，マウスで画面上をドラッグするだけで，ラベル，テキストボックス，チェックボックスといった様々な Windows 部品を容易に作成することができる。

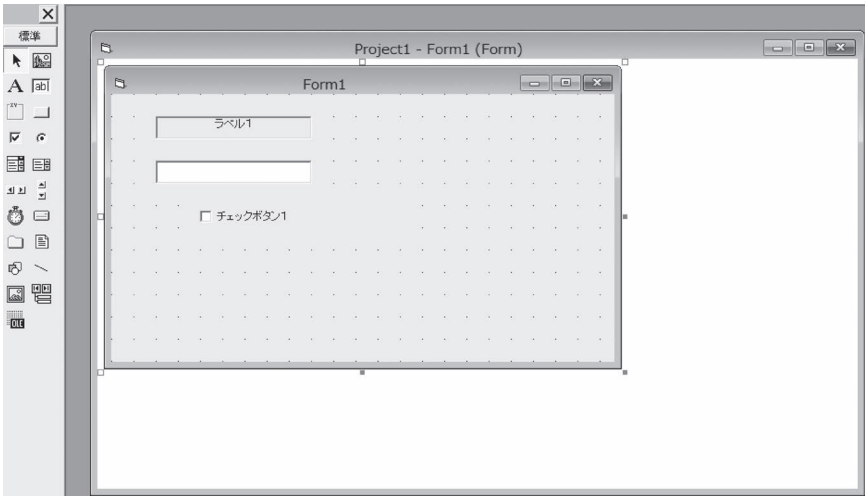

図 1　Visual BASIC 6.0 の Windows 部品作成例

もちろん，Visual BASIC にもビジュアル化されていない部分が存在する。Visual BASIC は，ビジュアルに，マウスを使って Windows 部品を作成することができるが，その Windows 部品に入力があった場合に行うべきロジックを記述する部分，すなわち，イベント駆動の部分の動作については，テキストプログラミング言語と同じく，テキストエディタでプログラミングするようになっている。その意味では，筆者は，Visual BASIC は不完全な半ビジュアル言語であると考える。

Ⅱ．　完全なビジュアル言語の設計・開発

1.　完全なビジュアル言語の要件

筆者は，様々なプログラミング言語を用いてソフトウェアを開発してきた。機械語，アセンブラ言語，BASIC，FORTRAN，COBOL，C，C++，Delphi，Visual BASIC を使用する中で，テキストプログラミングから脱却し，Visual BASIC のような半ビジュアル言語ではなく，完全なビジュアル言語を設計することはできないだろうかと考えた。

筆者の考える完全なビジュアル言語とは，Visual BASIC にように，画面上の部品を作成する部分をビジュアル化するだけでなく，これまではテキストエディタで記述していたロジックの部分のプログラミングもビジュアル化する言語を指す。BBB は，このような考え方を元に機能設計を行ったプログラミング言語である。

2. 完全なビジュアル言語実現の手法の提案：ボックスベース

完全なビジュアル言語を実現するためには，テキストエディタでない，ロジックの入力方法が必要となる。そこで，全ての操作を，画面上に「ボックス」と呼ぶ長方形を作成し，これを「ベース」とする「ボックスベース」の考え方により実現する手法を提案する。

そして，「ボックス」は，単純な長方形に，縦横に移動可能な分割線を設け，最大 4 分割する構造とする。これにより，複雑な情報をビジュアルに表示することが可能となる。

3. ロジックプログラミング部分の基本要素

今回，ロジックプログラミングに必要な基本要素を，以下の13個に限定した。これらの要素を組み合わせることにより，ビジュアルにロジックの部分をプログラミングすることが可能である。

A） 処理：ロジックの集合体（変数域と内容域を持つ）
B） 変数：動作中に値を保持する領域（定数の機能も含める）
C） 文字列：文字列（数値文字列，演算子も含める）
D） 計算式：変数に操作を加える式
E） 条件式：反復や分岐を行うときの真偽を評価するための式
F） 入力：外部からのデータ入力を行う文
G） 出力：外部へデータ出力を行う文
H） 反復：条件式に従って真偽を評価し，反復して計算式を実行する部分
I） 分岐：条件式に従って真偽を評価し，計算式を実行する部分
J） 脱出：反復から脱出する文
K） 終了：処理を終了する文
L） 呼出：他の処理を呼び出す文
M） 復帰：処理を終え，呼出元に制御を戻す文

4. ボックスへの操作（移動，拡大，縮小）

ボックスは，マウスでクリックして選択すると，図 2 に示すように，その周囲に 8 つの小さな箱を表示するようにしている。左上の[+]の箱は「移動」，その他の[ ]の箱は拡大・縮小を意味する。そして，マウスカーソルをこれらの小さな箱に重ねると，マウスカーソルが図 2 のように変化し，その状態で左ボタンを押してマウスを動かすと，「移動」「拡大・縮小」の操作が行われる。

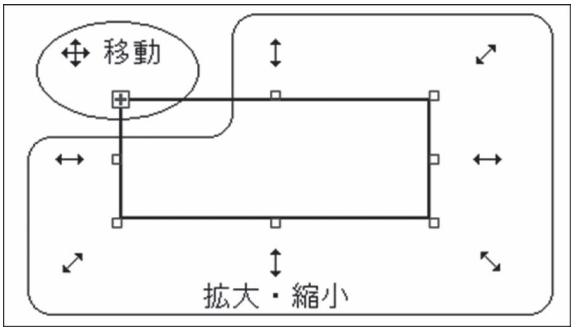

図 2　ボックスの「移動」「拡大・縮小」インタフェース

このような，マウスを使った「移動」「拡大・縮小」インタフェースは，一般的な Windows の方式を踏襲しながら，左上の位置を「移動」に割り当てることにより，ユーザにとって，必要な操作が即座に判断できる方式にしている。

5.　ボックスへの親子の概念の導入

すべてのボックスには，親子関係を設定する。そして，親となるボックスをマウスでクリックして選択し，移動すると，親の上に乗っている複数の子ボックスは，親ボックスに乗ったまま移動する。

例えば，図 3 のように，計算式という親ボックスの上に「SUM」「＝」「SUM」「＋」「X」という 5 つの子ボックスが乗っている場合，親ボックスを移動すると，5 つの子ボックスは，親ボックスに乗ったまま移動する。

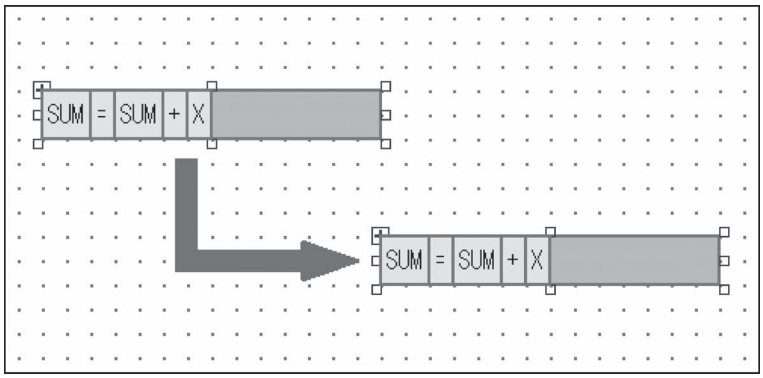

図 3　親ボックスの移動に伴う子ボックスの移動

次に，図 4 のように，計算式という親ボックスの上に「SUM」「＝」「SUM」「＋」「X」という 5 つの子ボックスが乗っていて，子ボックスである「X」と「SUM」を移動した場合，親ボックスの中で，子ボックスの位置や順序を変更することができる。

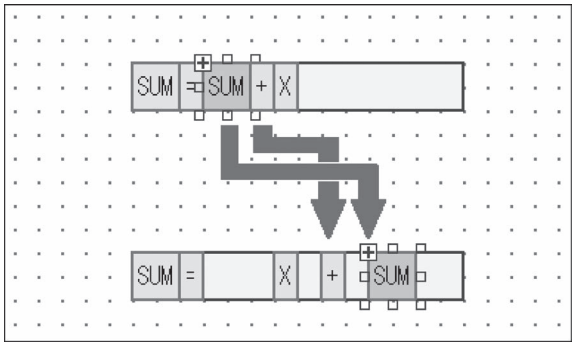

図 4　親ボックス上での子ボックス移動

6. 他のプログラミング言語ソース出力機能

BBB は，ネイティブモードでの実行のほかに，複数のプログラミング言語のソースを出力する機能を持たせている。BBB は，プログラミングを行うときの抽象度が高いため，他のプログラミング言語のソースを容易に出力することが可能である。BBB は，多くのプログラミング言語の中から，「N88-BASIC(86)」[3]，「i99-BASIC」[4]，「C 言語」の 3 言語をサポートしている。これにより，Windows パソコンだけでなく，UNIX ワークステーションや，FA パソコンといった，多くの環境の下で，BBB でプログラミングしたプログラムを動作させることが可能となる。

Ⅲ. BBB を用いたプログラミングの例

1. プログラミングサンプルの内容

BBB を用いたプログラミングのサンプルとして，「1 から100までの数を合計し，それを出力する」という内容で説明する。合計は，1 + 2 + 3 + 4 +…+99+100の式で求まる。

2. BBB の初期画面

BBB を起動すると，図 5 のような初期画面が表示される。

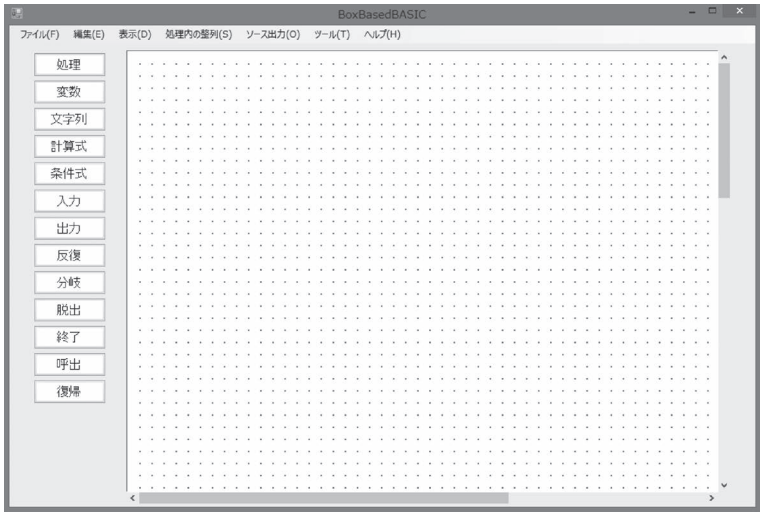

図 5 BBS の初期画面

3. 「処理」ボックスの作成

BBB の左に位置するボタンの中で，「処理」をクリックする。すると，「処理」ボタンが赤くなるので，ドットの表示された領域上で，マウスを左クリックしながら右下に移動させ，「処理」のボックスを作成する。その結果，図 6 のようになる。

4. 「処理」ボックスの処理名，概要の設定

「処理」ボックスで，4 分割されたボックスの左上部分をクリックする。すると，図 7 のよう

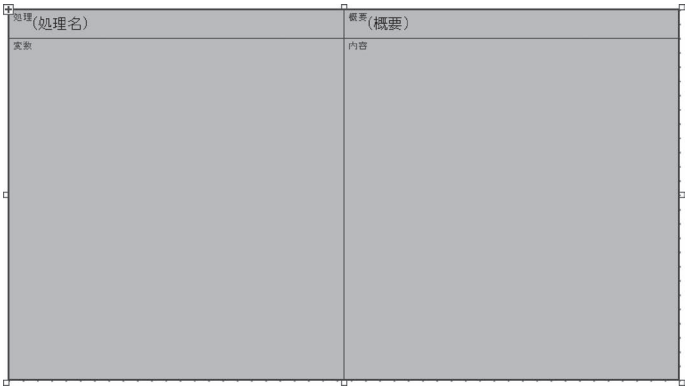

図 6　「処理」ボックスの作成

図 7　「入力・変更画面」の作成

な「入力・変更画面」が表示され，デフォルトの（処理名）という文字列が表示されるので，これを消去して「集合計算」に変更する。

同時に，4 分割されたボックスの右上部分をクリックする。すると，「入力・変更画面」が表示され，デフォルトの（概要）という文字列が表示されるので，これを消去して「1 から100までの数を合計する」に変更する。その結果，図 8 のように，ボックス内に文字列が表示される。

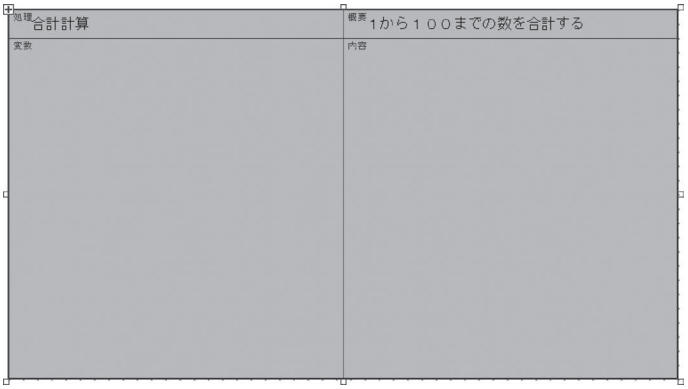

図 8　「処理」ボックスの処理名・概要変更結果

5.　「反復」ボックスの作成

BBB の左に位置するボタンの中で，「反復」をクリックする。すると，「反復」ボタンが赤く

なるので，「処理」ボックスの右下の領域上で，マウスを左クリックしながら右下に移動させ，「反復」のボックスを作成する。その結果，図９のように，「処理」ボックスの中に「反復」ボックスが表示される。

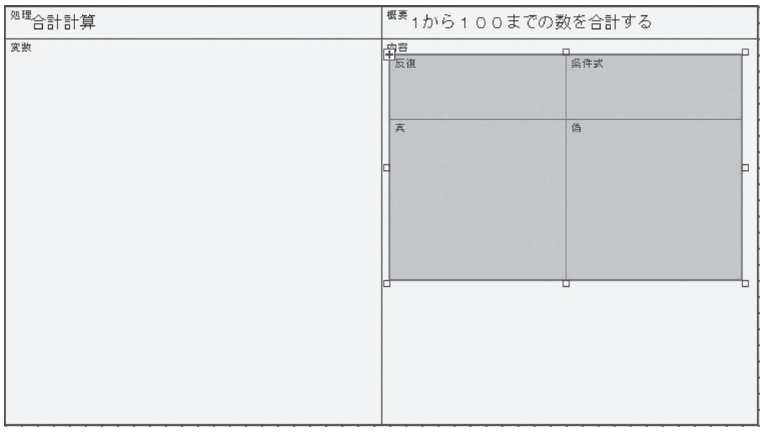

図９ 「反復」ボックスの作成

6. 「反復」ボックスの「条件式」部分への「条件式」の作成

BBB の左に位置するボタンの中で，「条件式」をクリックする。すると，「条件式」ボタンが赤くなるので，「反復」ボックスの右上の「条件式」の領域上で，マウスを左クリックしながら右下に移動させ，「条件式」のボックスを作成する。その結果，図10のように，「反復」ボックス上に「条件式」が表示される。

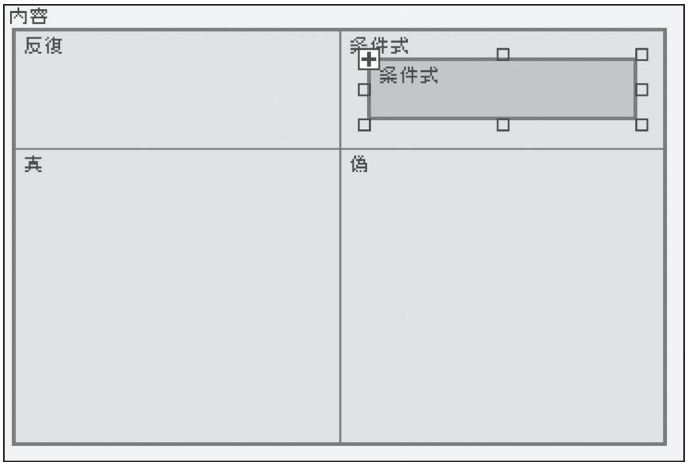

図10 「反復」ボックス右上への「条件式」の作成

7. 「条件式」ボックスの入力

「反復」ボックス右上の「条件式」をダブルクリックする。その結果，図11のような「条件式」入力・変更画面が表示されるので，「X＜＝C」と入力する。

図11 「入力・変更画面」の作成

X は 1 から100まで数えるカウンタ変数，C は100（回）という定数である。X が100以下の間，真の処理を行うことを意味する。図12に，「条件式」ボックスへの条件式入力作成後の画面を示す。なお，BBB は自動的に，「処理」ボックスの変数域に，「X」と「C」という 2 つの変数を追加する。初期値はそれぞれ「 0 」の設定にしてある。

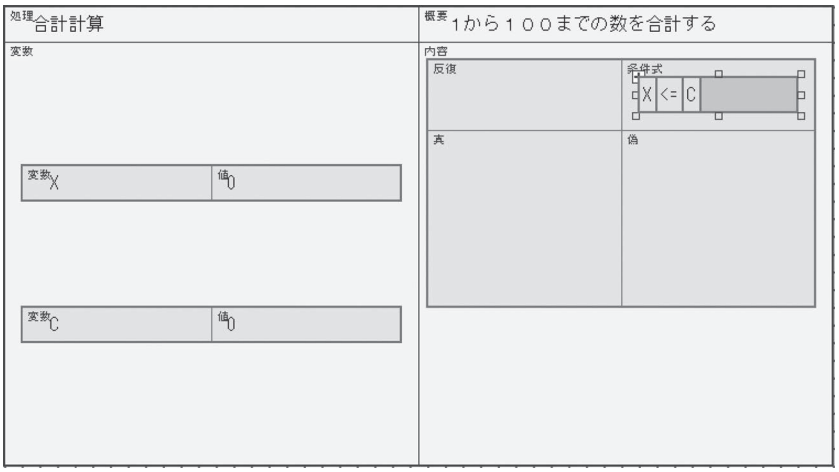

図12 「条件式」ボックスへの条件式入力作成後

8. 「処理」の変数への値の設定

BBB が「処理」ボックスの変数域に自動的に作成した，「X」と「C」という 2 つの変数に，初期値を設定する。「X」に 1 ，「C」に100を設定すると，図13のようになる。

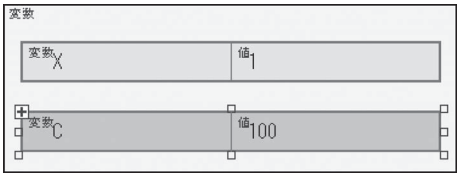

図13 「変数」への値の入力後

9. 「反復」の「真」の領域への「計算式」の設定

BBB の左に位置するボタンの中で，「計算式」をクリックする。すると，「計算式」ボタンが

赤くなるので，「反復」ボックスの左下の「真」の領域上で，「SUM＝SUM＋1」「X＝X＋1」という２つの計算式を作成する。その結果を図14に示す。

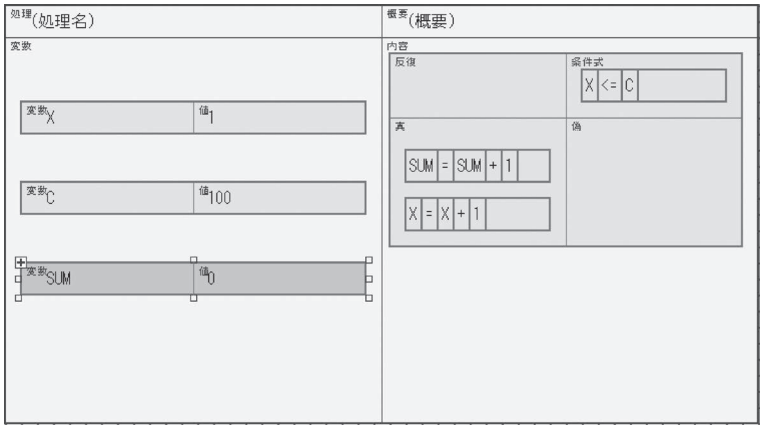

図14 「反復」の「真」の領域への計算式の設定

10. 「反復」の「偽」の領域への「脱出」の設定

BBB の左に位置するボタンの中で，「脱出」をクリックする。すると，「脱出」ボタンが赤くなるので，「反復」ボックスの右下の「偽」の領域上で，「脱出」ボックスを作成する。その結果を図15に示す。

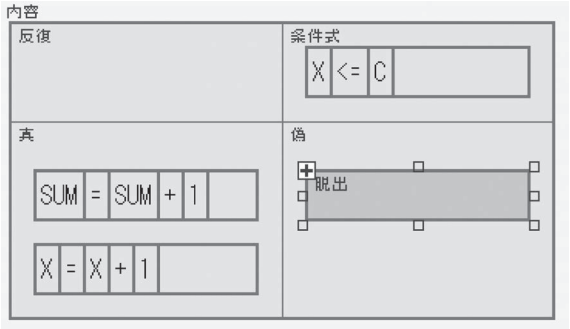

図15 「反復」の「偽」の領域への「脱出」ブロックの設定

11. 「処理」の「内容」の領域への「出力」ボックスの設定

BBB の左に位置するボタンの中で，「出力」をクリックする。すると，「出力」ボタンが赤くなるので，「処理」ボックスの右下の「内容」の領域上で，「出力」ボックスを作成する。そして，「出力」ボックスをダブルクリックし，「１～100の合計は “SUM” です。」と入力する。“SUM” は，出力文の中で，「SUM」が変数であることを示す。その結果を図16に示す。

12. 「処理」の「内容」の領域への「終了」ボックスの設定

BBB の左に位置するボタンの中で，「終了」をクリックする。すると，「終了」ボタンが赤く

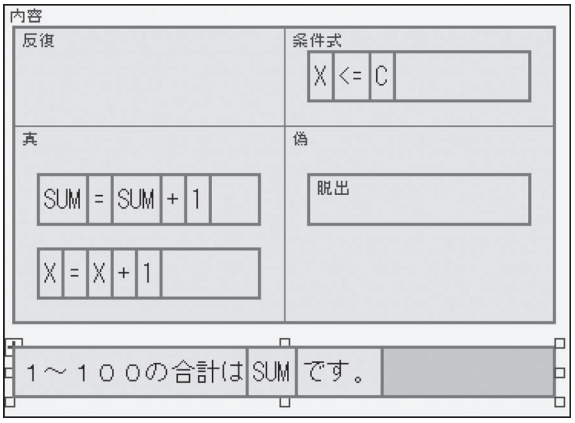

図16 「処理」の「内容」への「出力」ブロックの設定

なったので,「処理」ボックスの右下の「内容」の領域上で,「終了」ボックスを作成する。その結果を図17に示す。

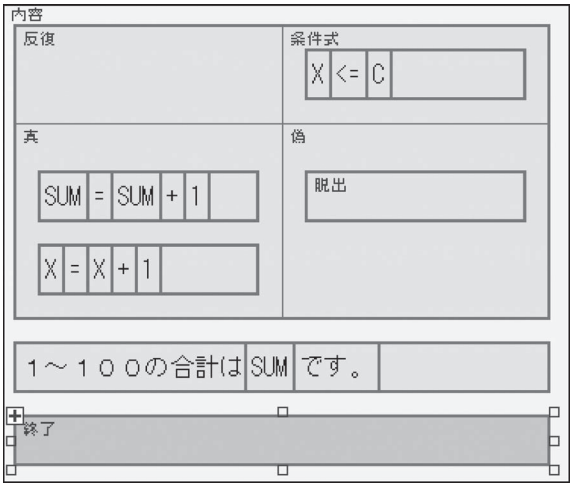

図17 「処理」の「内容」への「終了」ブロックの設定

13. BASIC ソースの出力

BBB のメニュー「ソース出力(O)」の「N88-BASIC(86)(N)」を選ぶ。その結果，N88-BASIC のソースプログラムが作成される。図18に，出力されたソースを示す。

14. N88 互換 BASIC for Windows95 上での実行

N88 互換 BASIC for Window95[5] を実行し，BBB で作成した BASIC ソースを読ませると，図19のようになる。

そして，実行ボタンにより実行した結果，図20のような実行結果が表示される。

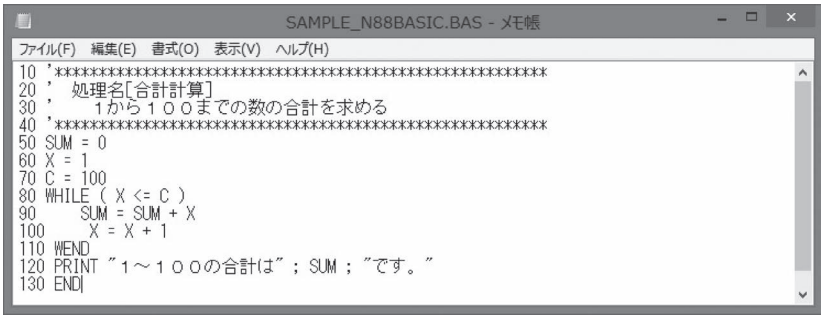

図18　N88-BASIC ソース

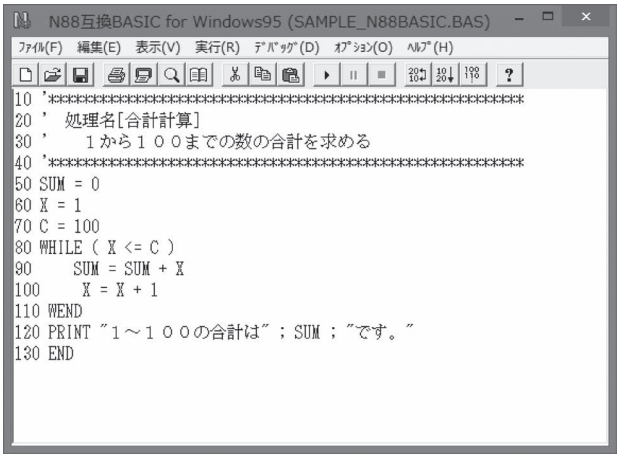

図19　N88互換 BASIC forWindows95 上での読込

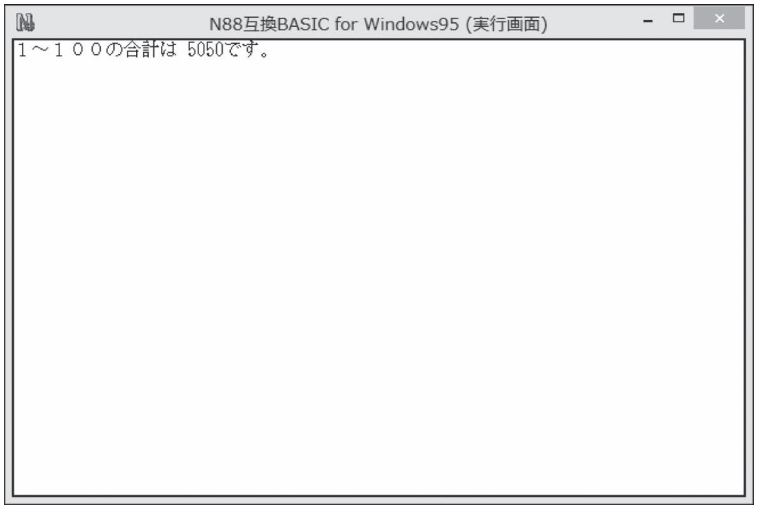

図20　N88互換 BASIC forWindows95 上での実行結果

Ⅳ. BBB の技術的な特徴

1. 構造化プログラミング手法における順次・反復・分岐の直接的実現

構造化プログラミング手法[6] においては，３つの構造化文によるプログラミングが推奨される。それは，順次，反復，分岐という３つの構造化文である。

順次は，プログラムに記されたとおりに実行すること，反復は，一定の条件が満たされている間処理を繰り返すこと，分岐は，条件に応じて別の処理に分かれることを意味する。

BBB は，これら３つの構造化文を，直接的に実現するように設計した。順次については，計算式ボックスが置かれた位置の通りに実行されることで実現し，反復は反復ボックス，分岐は分岐ボックスを実装することで実現した。また，反復から脱出するために脱出ボックスを実装した。

このように，直接的に構造化プログラミング手法によりプログラミングが行えることが，BBB の最大の特徴である。

2. 変数管理の自動化

プログラミング言語において，変数名の管理は重要である。変数を宣言せずに使用できる言語も存在するが，多くの言語においては，変数を宣言した後に使用する必要があり，プログラマにとって，そのことが大きな負担である。

BBB では，そのような負担を軽減するために，変数管理を自動化した。例えば，プログラマが計算式や条件式の中で変数を使用した場合，同名の変数宣言を行なっていなければ，変数を宣言するボックスを自動的に生成する機能を持っている。これにより，変数名の重複や宣言忘れを防止することができる。

また，自動生成した変数が，その後の計算式変更等により，不要になることがある。そのような場合，BBB のメニューの中に設けた「未使用変数の削除」を実行すると，不要な変数宣言を削除する機能を持たせており，無駄な変数宣言が放置されることを防止することができる。

このように，変数管理を自動化することにより，プログラマの負担を軽減できることが，BBB の大きな特徴である。

3. 複数プログラミング言語のソース出力機能

⑴ 「N88-BASIC(86)」「i99-BASIC」「C 言語」のソース出力機能

BBB は，ネイティブモードでの実行のほかに，複数のプログラミング言語のソースを出力する機能を有する。現在，「N88-BASIC(86)」「i99-BASIC」「C 言語」の３言語をサポートしている。図21に，３つの言語で出力したソースの比較を示す。

BBB 自体は，ボックスを作成し，それらの組み合わせでプログラミングしており，他のプログラミング言語のソースを出力することは容易である。

⑵ 比較演算子の違い

BBB がソースを出力するプログラミング言語は，BASIC と C 言語である。この２つの言語の文法は，微妙に異なる部分が存在する。例えば，否定を意味する比較演算子は，BASIC の場合「<>」であるが，C 言語の場合「!=」となる。等価を意味する比較演算子は，BASIC の場合「=」であるが，C 言語の場合「==」となる。このように，２言語間で演算子が異なる場合，C 言語で出力する時点で，プログラマが記述した BASIC の演算子から C 言語の演算子に強制的に変換し，

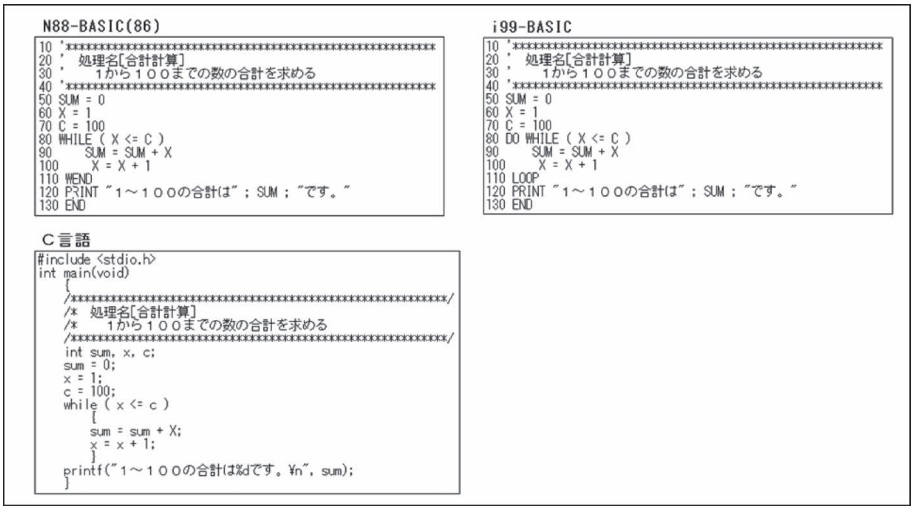

図21 3言語でのソース出力比較

言語間の違いを吸収している。

(3) 反復機能の違い

BBB がソースを出力するプログラミング言語の仕様の違いにより，反復の機能を実現する際に工夫をする必要がある。

(ア) N88-BASIC(86)

FOR ～ TO ～ STEP ～ NEXT，WHILE ～ WEND の2つのみが，反復を実現する命令である。そのため，BBB の「反復」ブロックは，WHILE ～ WEND に変換している。

(イ) i99-BASIC

FOR ～ TO ～ STEP ～ NEXT，DO ～ LOOP，DO ～ LOOP UNTIL，DO ～ LOOP WHILE，DO UNTIL ～ LOOP，DO WHILE ～ LOOP，WHILE ～ WEND と，反復を実現する多くの命令が用意されている。そのため，BBB の「反復」ブロックは，DO UNTIL ～ LOOP，DO WHILE ～ LOOP に変換している。

(ウ) C 言語

for 文，while 文，do while 文と，反復を実現する3つの命令が用意されている。そのため，BBB の「反復」ブロックは，while 文に変換している。

(4) 条件分岐の違い

BBB で，条件分岐を含む式をプログラミングするとき，どの言語に出力するかによって，ソースに大きな違いが生じる。

図22は，2つの数を入力し，「和」か「差」と入力すると，2つの数の和か差を出力するプログラムを BBB で記述したものである。

N88-BASIC では，IF ～ ELSE ～ ENDIF が使えない。そのため，GOTO 文を使って処理を分けざるを得ない。しかし，i99-BASIC では，IF ～ ELSE ～ ENDIF が使えるため，GOTO 文なしで同じ機能を実現できる。C 言語では，if ～ else ～に変換する。

図23に，N88-BASIC 用に作成したソースを，図24に，i99-BASIC 用に作成したソースを示す。

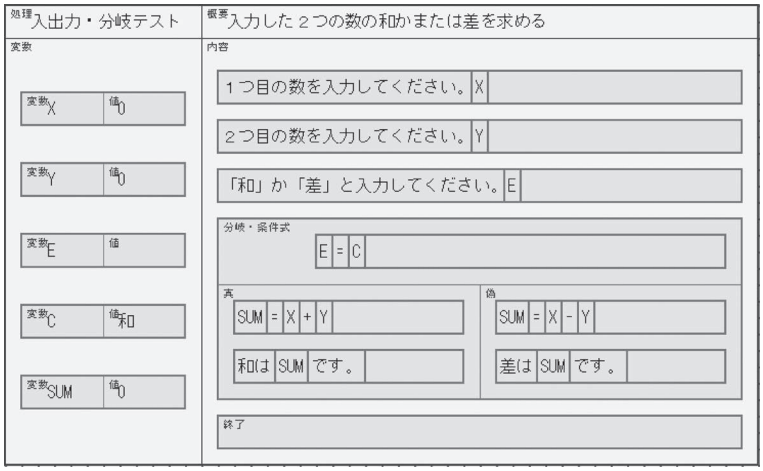

図22　２つの数の和か差を求めるプログラム（BBB）

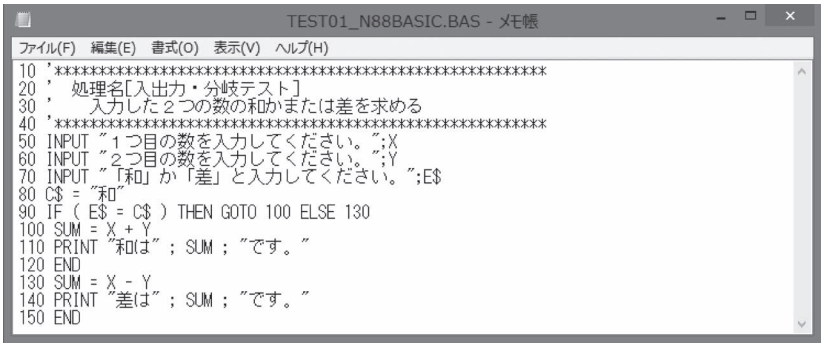

図23　N88-BASIC 用の GOTO 文で実現したソース

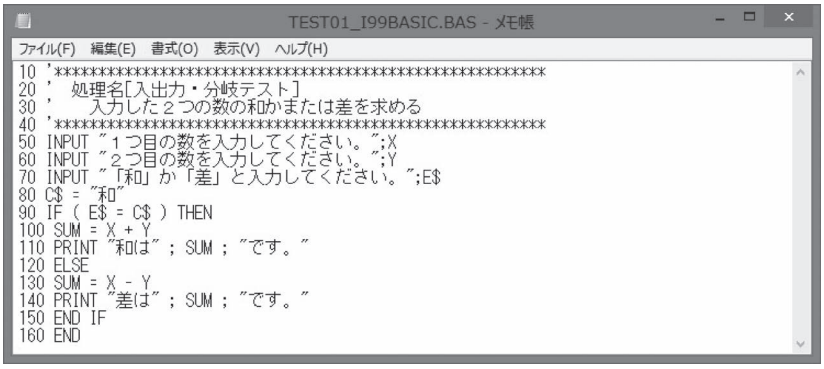

図24　i99-BASIC 用の GOTO 文なしで実現したソース

結 論

BBB は，プログラミング言語における完全なビジュアル化を実現することを目標に設計を行ったプログラミング言語である。BBB は，画面上の部品を作成する部分をビジュアル化するだけでなく，これまではテキストエディタで記述していたロジックの部分のプログラミングも含め，全ての操作を，画面上に作成する「ボックス」と呼ぶ長方形によってビジュアル化した。この「ボックス」は，最大 4 分割できる構造とすることにおり，複雑な情報をビジュアルに表示することが可能となった。また，BBB に必要な基本要素を13個に限定し，それらの要素を組み合わせることにより，完全なビジュアル化を実現した。

BBB は，構造化プログラミング手法を理解するときに大変役に立つ。それは，構造化プログラミング手法における順次・反復・分岐の機能を，BBB が直接的に実現しているからである。また，変数管理の自動化により，これまでプログラマの大きな負担となっていた変数について，その負担を減らすことに成功したり，複数プログラミング言語へのソース出力機能を持たせることにより，Windows 搭載パソコンから FA 用パソコンまで，広範囲の環境で動作させることができるようになるなど，多くの利点を有する。

筆者は，BBB とは別に，i99-BASIC の GUI 部の開発を容易にするために，i99-BASIC Assist というソフトウェアを開発した[7]。これは，マウスを用い，i99-BASIC がサポートする部品を作成するツールであり，GUI 部の開発を極めて短時間で行うことができるようになった。BBB の設計には，i99-BASIC Assist のアイデアを多く流用している。

Visual BASIC が，ビジュアルなプログラミング手法の道を開き，多くのプログラマを惹きつけたように，ロジックの部分のプログラミングもビジュアルに行える BBB の機能をさらに拡充し，多くのプログラマに使ってもらえるようなプログラミング言語に成長させていきたいと考える。

注

1) Visual BASIC は，Microsoft 社が販売したプログラミング開発環境である。当初は MS-DOS 環境下でのビジュアルなプログラミングツールとして開発された。その後，1995年に Windows95 が発売され，Windows アプリケーションソフトウェアを手軽に開発できるプログラミング言語として注目され，多くの Windows アプリケーションが Visual BASIC によって開発された。特に，1998年に発売された Visual BASIC 6.0 は，データベースアクセス機能，ActiveX 作成機能など，高度なプログラミングも可能となり，高い評価を得た。
2) 吉田弘一郎（1994），「Visual C++ による MFC ライブラリの使い方」参照。
3) N88-BASIC（86）は，NEC 社のパソコン PC-9801 シリーズ上に搭載されていたプログラミング開発環境である。N88-BASIC（86）は，手軽に MS-DOS 上で動作する BASIC プログラムを開発できたため，多くのユーザを獲得していた。その後，Windows の普及により，Windows 上では動作しない N88-BASIC（86）は急速に廃れてしまった。
4) i99-BASIC は，インタフェース社が発売している産業用コンピュータ上を制御するプログラミング言語である。制御，計測といった産業用コンピュータ上に必須の機能を，BASIC 言語により容易に行えることがその特徴である。i99-BASIC は，制御・計測を行う前の段階で，プログラミングに多くの時間を割いていた技術者にとって，プログラミング時間を大幅に短縮できる利点を有する。
5) N88互換 BASIC for Window95は，潮田康夫氏が開発した，Windows 上で動作する N88-BASIC（86）プログラミング開発環境である。N88互換 BASIC for Window95は，N88-BASIC（86）のソースを，そのまま Windows 上で動作させることのできる貴重な開発環境となっている。
6) E. W. Dijkstra, “Structured Programming”, In Software Engineering Techniques, B. Randell and J. N.

Buxton, (Eds.), NATO Scientific Affairs Division, Brussels, Belgium, 1970, pp. 84–88参照。
7） 山下明博（2013），「『i99-BASIC Assist』の開発：『i99-BASIC』による GUI ソフトウェア開発の支援ツール」，2012年度安田女子大学現代ビジネス学会誌。

〔2013. 9．26 受理〕